Data Structures and Algorithms

Lecture 18

Aniket Basu Roy

BITS Pilani Goa Campus

2026-03-02 Mon

Agenda

- Data Structures
- Hash Tables
- Chaining
- Hash Functions

The Dictionary Problem

- Given: A dynamic set
- We want to support:
  - INSERT
  - DELETE
  - SEARCH

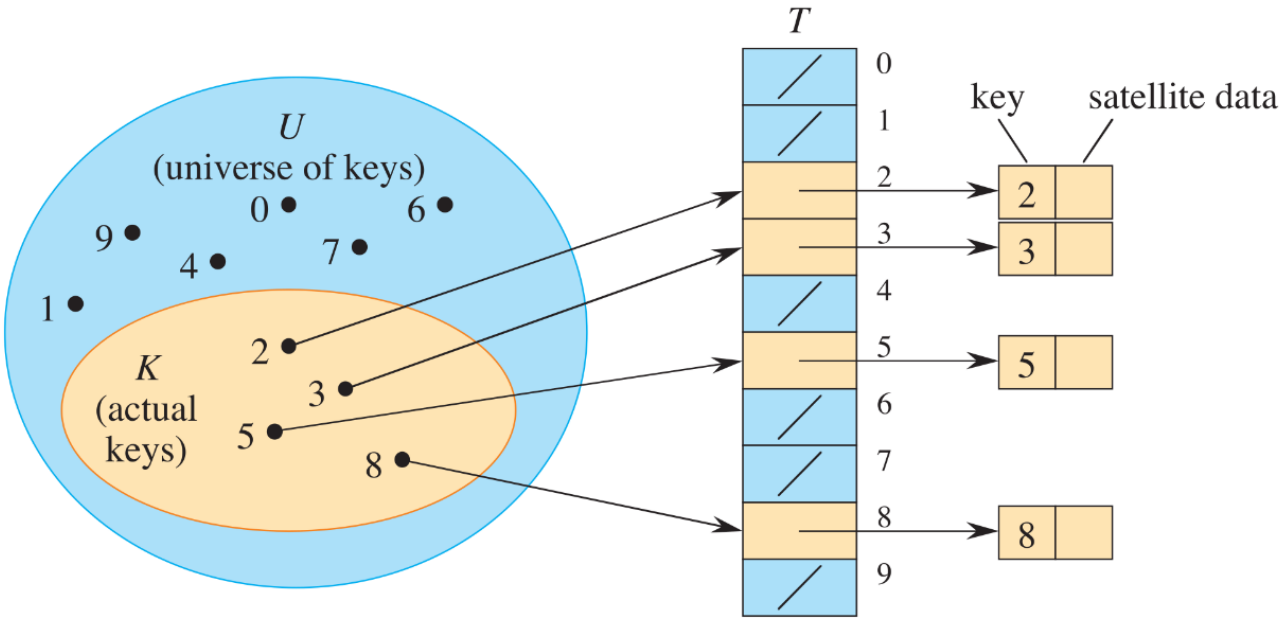
The Dictionary Problem

- Given: A dynamic set
- We want to support:
  - INSERT
  - DELETE
  - SEARCH
- Keys from a universe $U = \{0, 1, \ldots, |U| - 1\}$.

The Dictionary Problem

- Given: A dynamic set
- We want to support:
  - INSERT
  - DELETE
  - SEARCH
- Keys from a universe $U = \{0, 1, \ldots, |U| - 1\}$.
- Goal: $O(1)$ expected time per operation.

The Dictionary Problem



Direct-Address Tables

Allocate array $T[0 : |U| - 1]$. Slot $T[k]$ holds element with key k.

```
DIRECT-ADDRESS-SEARCH(T, k)    return T[k]
DIRECT-ADDRESS-INSERT(T, x)    T[x.key] = x
DIRECT-ADDRESS-DELETE(T, x)    T[x.key] = NIL
```

Direct-Address Tables

Allocate array $T[0 : |U| - 1]$. Slot $T[k]$ holds element with key k.

```
DIRECT-ADDRESS-SEARCH(T, k)    return T[k]
DIRECT-ADDRESS-INSERT(T, x)    T[x.key] = x
DIRECT-ADDRESS-DELETE(T, x)    T[x.key] = NIL
```

All operations: $\Theta(1)$.

Direct-Address Tables

Allocate array $T[0 : |U| - 1]$. Slot $T[k]$ holds element with key k.

```
DIRECT-ADDRESS-SEARCH(T, k)    return T[k]
DIRECT-ADDRESS-INSERT(T, x)    T[x.key] = x
DIRECT-ADDRESS-DELETE(T, x)    T[x.key] = NIL
```

All operations: $\Theta(1)$.
Problem: If $|U|$ is large, T requires $\Theta(|U|)$ space — infeasible.

Hash Tables: The Idea

Use a hash function $h : U \to \{0, 1, \ldots, m-1\}$.

Hash Tables: The Idea

Use a hash function $h : U \to \{0, 1, \ldots, m-1\}$.

- Element with key k goes in slot h(k).

Hash Tables: The Idea

Use a hash function $h : U \to \{0, 1, \ldots, m-1\}$.

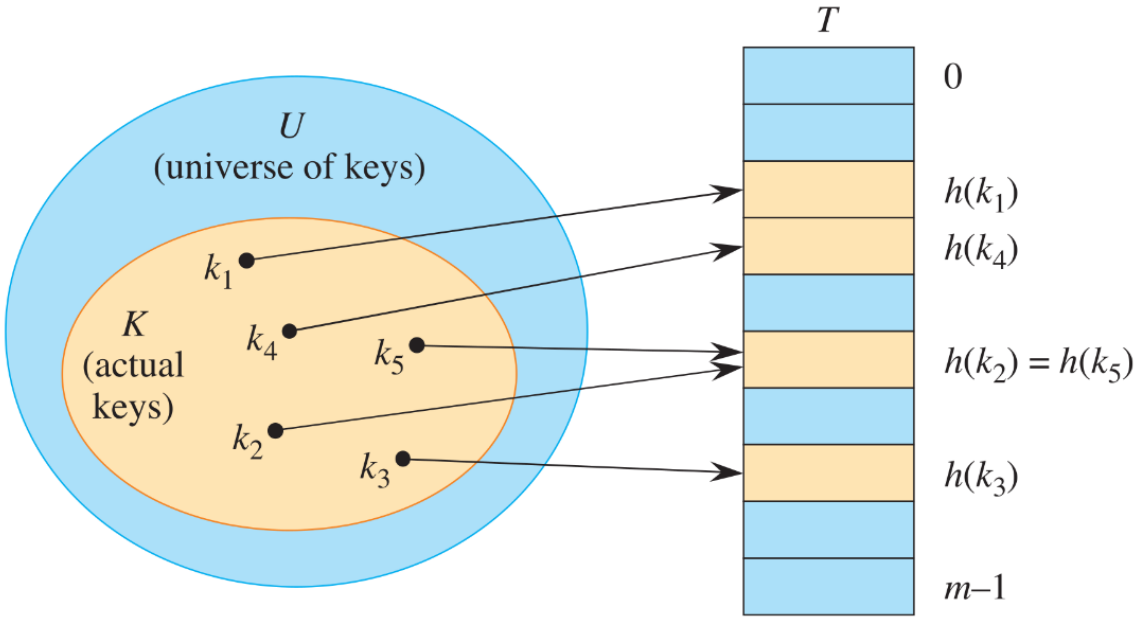
- Element with key k goes in slot h(k).
- Table size $m \ll |U|$ $\Rightarrow$ space $\Theta(m)$.

Hash Tables: The Idea

Use a hash function $h : U \to \{0, 1, \ldots, m-1\}$.

- Element with key k goes in slot h(k).
- Table size $m \ll |U|$ $\Rightarrow$ space $\Theta(m)$.
- Problem: Two distinct keys may map to the same slot, aka collision.

Hash Tables: The Idea



Chaining

Store all elements hashing to slot j in a linked list at $T[j]$.

```
T
0 --> NIL
1 --> [49] --> [86] --> NIL
2 --> [17] --> NIL
3 --> NIL
4 --> [52] --> [30] --> NIL
```

Chaining

```
CHAINED-HASH-INSERT(T, x)
    insert x at head of T[h(x.key)]

CHAINED-HASH-SEARCH(T, k)
    search list T[h(k)] for key k

CHAINED-HASH-DELETE(T, x)
    delete x from list T[h(x.key)]
```

Load Factor α

$$\alpha = \frac{n}{m}$$

The average length of each chain.

Load Factor α

$$\alpha = \frac{n}{m}$$

The average length of each chain.

- Simple Uniform Hashing Assumption (SUHA):

Each key equally likely to hash to any slot, independently.

Load Factor α

$$\alpha = \frac{n}{m}$$

The average length of each chain.

- Simple Uniform Hashing Assumption (SUHA):

Each key equally likely to hash to any slot, independently.

- Under SUHA, expected chain length at any slot $= \alpha$.

Expected Search Times

Under SUHA:

Expected Search Times

Under SUHA:

Unsuccessful search: examine entire chain at $T[h(k)]$.

$$\text{Expected cost} = \Theta(1 + \alpha)$$

Expected Search Times

Under SUHA:

Unsuccessful search: examine entire chain at $T[h(k)]$.

$$\text{Expected cost} = \Theta(1 + \alpha)$$

Successful search: examine on average half the chain beyond target.

$$\text{Expected cost} = \Theta(1 + \alpha)$$

- (See the lecture outline for the derivation.)

Expected Search Times

Under SUHA:

Unsuccessful search: examine entire chain at $T[h(k)]$.

$$\text{Expected cost} = \Theta(1 + \alpha)$$

Successful search: examine on average half the chain beyond target.

$$\text{Expected cost} = \Theta(1 + \alpha)$$

- (See the lecture outline for the derivation.)

Key Takeaway

If $n = O(m)$ (i.e., $\alpha = O(1)$), all operations take $O(1)$ expected time.